Лабораторна робота №3

Для виконання проекту з ООП ми використовували платформу JavaFX, яка дозволяє створювати графічний інтерфейс користувача.

Перевагою JavaFX є можливість швидко та якісно створити графічний інтерфейс, що дуже корисно, коли треба створити проект в короткий термін. Наприклад, JavaFX містить у собі різні види кнопок, полів, меню та навіть діаграм. Також JavaFX використовує стиль CSS і для графічного інтерфейсу ми можемо використовувати файли спеціального формату FXML замість того, щоб писати все в Java коді.

Встановити та почати використовувати цю платформу виявилось доволі просто, середовище IntelliJ IDEA саме запропонувало встановити потрібні файли, також довелося скачати Scene Builder. В інтернеті про JavaFX до волі багато інформації, тож розібратися з основними моментами виявилось нескладно (дуже сильно допомогло відео [JavaFX GUI Course ☕ (𝙁𝙧𝙚𝙚) - YouTube](https://www.youtube.com/watch?v=9XJicRt_FaI)). Поведінка класів та методів виявилась більш-менш зрозумілою

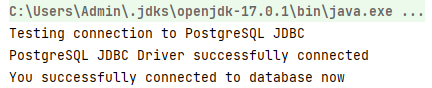
Одним із не дуже приємних аспектів можна назвати те, що деякі спеціалісти називають JavaFX застарілою і навіть є статі із заголовками «JavaFX мертвий чи живий?» ([JavaFX мёртв или жив? Анализ рынка. Аналитический блог разработчика. Заметки, идеи, мысли (initkfs.github.io)](https://initkfs.github.io/pages/javafx-dead-or-alive.html)). Звісно, не можна казати що платформу ніде нині не використовують, але сам факт існування таких статей змушує задуматись.

Як інший ресурс, який можливо сюди віднести – це підключення бази даних [Postgresql](https://www.postgresql.org/). Ось як це виглядає в *Intellij IDEA*:

* спершу код тестової міні-програми, яка перевіряє чи справді нашу програму підключено до БД:

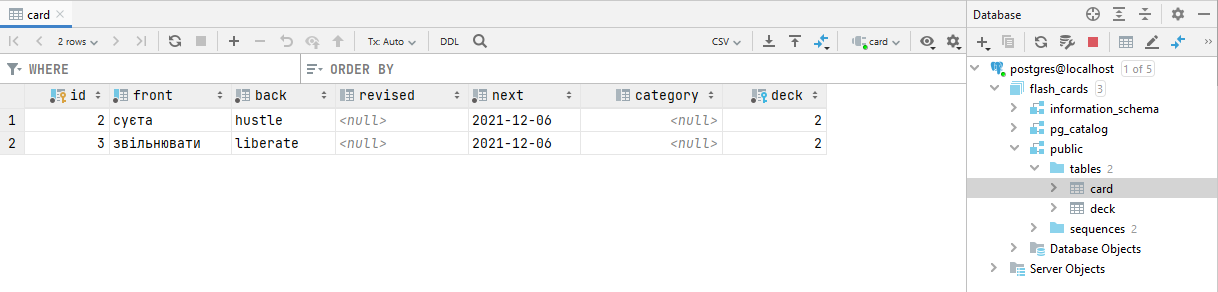
**package** com.cybernetics.flashcardsdemo;  
  
**import** java.sql.Connection;  
**import** java.sql.DriverManager;  
**import** java.sql.SQLException;  
  
**public class** DBConnection {  
 *// Database credentials* **static final** String ***DB\_URL*** = **"jdbc:postgresql://localhost:5432/postgres"**;  
 **static final** String ***USER*** = **"postgres"**;  
 **static final** String ***PASS*** = **"19Max#177p"**;  
  
 **public static void** main(String[] argv) {  
  
 System.***out***.println(**"Testing connection to PostgreSQL JDBC"**);  
  
 **try** {  
 Class.*forName*(**"org.postgresql.Driver"**);  
 } **catch** (ClassNotFoundException e) {  
 System.***out***.println(**"PostgreSQL JDBC Driver is not found. Include it in your library path "**);  
 e.printStackTrace();  
 **return**;  
 }  
  
 System.***out***.println(**"PostgreSQL JDBC Driver successfully connected"**);  
 Connection connection;  
  
 **try** {  
 connection = DriverManager  
 .*getConnection*(***DB\_URL***, ***USER***, ***PASS***);  
  
 } **catch** (SQLException e) {  
 System.***out***.println(**"Connection Failed"**);  
 e.printStackTrace();  
 **return**;  
 }  
  
 **if** (connection != **null**) {  
 System.***out***.println(**"You successfully connected to database now"**);  
 } **else** {  
 System.***out***.println(**"Failed to make connection to database"**);  
 }  
 }  
}

Результат:

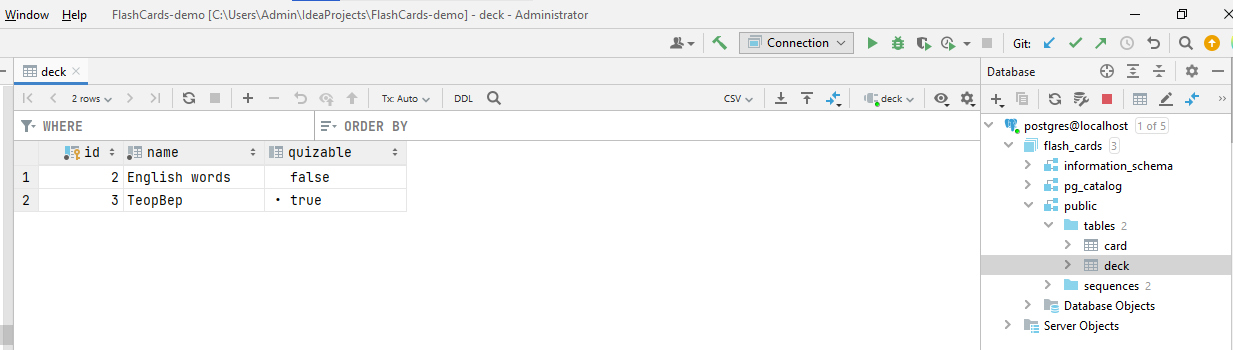


Тобто підключення пройшло успішно.

* також вже в самому середовищі розробки *Postgresql*, який називаєтья *pgAdmin*, було додано таблиці репрезентують дані нашої програми s і їх вже потім можна спокійно переглядати в *Intellij IDEA*, зокрема:
* таблиця *card*



* таблиця *deck*



Отже, як видно для зручного та ефективного збереження даних нашої програми було вирішено використовувати Реляційну базу даних Postgresql, яка на сьогоднішній день є найпотужнішою RDBMS, яка до того ще й безкоштовна.